Deep Learning for Image Sequence Classification of Astronomical Events

Rodrigo Carrasco
University of Chile
11 PUBLICATIONS **37** CITATIONS
SEE PROFILE

Francisco Förster
University of Chile
237 PUBLICATIONS **1,666** CITATIONS
SEE PROFILE

Pablo Estevez
University of Chile
156 PUBLICATIONS **2,723** CITATIONS
SEE PROFILE

Pablo Andres Huijse
Universidad Austral de Chile
35 PUBLICATIONS **278** CITATIONS
SEE PROFILE

Deep Learning for Image Sequence Classification of Astronomical Events

Rodrigo Carrasco Davis[1,7], Guillermo Cabrera-Vives[2,7], Francisco Förster[6,7], Pablo A. Estévez[1,7], Pablo Huijse[3,7], Pavlos Protopapas[5], Ignacio Reyes[1,7], Jorge Martínez[4,6,7] and Cristóbal Donoso[2]
[1]Dept. of Electrical Engineering, Universidad de Chile
[2]Dept. of Computer Science, Universidad de Concepción
[3]Informatics Institute, Universidad Austral de Chile
[4]Department of Astronomy, Universidad de Chile
[5]Institute for Applied Computational Science, Harvard University
[6]Center for Mathematical Modeling, Universidad de Chile
[7]Millennium Institute of Astrophysics, Chile

Abstract

We propose a new sequential classification model for astronomical objects based on a recurrent convolutional neural network (RCNN) which uses sequences of images as inputs. This approach avoids the computation of light curves or difference images. To the best of our knowledge, this is the first time that sequences of images are used directly for the classification of variable objects in astronomy. In addition, we solve partially the problem of transfer learning from synthetic to real-world images. This is done by transforming synthetic light-curves to images in a realistic way, by taking into account observational conditions and instrumental parameters. This approach allows us to generate datasets to train and test our RCNN model for different astronomical surveys and telescopes. Moreover, using a simulated dataset is faster and more adaptable to different surveys and classification tasks compared to collecting real labeled image sequences. To test the RCNN classifier trained with a synthetic dataset, we used real-world data from the High cadence Transient Survey (HiTS) obtaining an average recall of 87% on four classes: supernovae, RR Lyrae, non–variables, and asteroids. We compare the results of our model with those of a light curve classifier, in both simulated and real data. Our RCNN model outperforms the light curve classifier due to the extra information contained on the images. The results obtained encourage us to use and continue developing the proposed method for *astronomical alert brokers* systems that will process alert streams generated by new telescopes such as the Large Synoptic Survey Telescope and the Zwicky Transient Facility.

Keywords: astronomical databases: miscellaneous - methods: statistical - methods: data analysis - supernovae: general - techniques: image processing

1 INTRODUCTION

Astronomy is faced with the challenge of increasingly large streams of data produced by large survey telescopes. New telescopes, such as the Large Synoptic Survey Telescope (LSST, Ivezic et al., 2008) and the Zwicky Transient Facility (ZTF, Smith et al., 2014) are designed to study variables and transients on wide areas of the sky. Variable stars, such as pulsating (e.g. RR Lyrae, Cepheids) or eclipsing stars; or transients, such as supernova explosions, are expected to be produced in large numbers. These objects have characteristic timescales from hours to months, and can be detected and characterized by repeatedly observing the same region of the sky. Obtaining these repeated images with large cameras will generate a very large volume of data. For example, it is estimated that the LSST will generate 30 TB of raw data per night to produce a complete image of the southern sky every 3 days.

Some research areas in astronomy require the classification of a large number of objects: e.g. supernovae are needed to estimate cosmological distances for studies about the expansion of the universe (Riess et al., 1998; Schmidt et al., 1998), and variable stars such as RR Lyrae or Cepheids are needed to map the structure of the Milky Way and serve as *cosmic distance ladders* (Ngeow et al., 2013; Feast et al., 2014). In order to classify different astronomical objects in these large data streams we need to apply fast and accurate classification methods capable of managing large amounts of data in real-time. This problem will be addressed by systems called *astronomical alert brokers* (e.g. ALeRCE, ANTARES Narayan et al. (2018), LASAIR), which are capable of receiving, processing, classifying and reporting important information about the alert streams generated by these telescopes in real time.

Traditional methods to classify variable astronomical objects are based on pre-processing a sequence of images (calibration) followed by feature extraction (measurement). One way to extract features from a sequence of images is doing photometry (e.g. Naylor, 1998), which is the calculation of the total amount of light arriving from the source to the camera as a function of time, generating a time series called a *light curve*. In principle, a point–like source's light curve should contain all the relevant information about the source, but when the detection is spurious the information contained in the pixels becomes more relevant for the classification. Additionally, extragalactic sources such as supernovae tend to be near extended sources, i.e. galaxies, whereas galactic variable stars tend to be relatively isolated, information which is also contained in the image pixels.

Obtaining the light curve reliably requires performing difference imaging first for certain sources (e.g. when the object occurs in a bright galaxy), which is the process of aligning, convolving and differencing pairs of images to show only those pixels which have changed from frame to frame (e.g. Förster et al., 2016). Computing the difference image presents some problems, most of the time it is necessary to reduce the quality of one of the two images to subtract them correctly and it is also very sensitive to alignment errors between the frames.

Once the full light curves are computed, additional features can be extracted by manual design or automatic learning from the data. In the case of manual feature extraction, the scientist must design attributes that are expressive enough to contain relevant information for the classification, which could require a lot of effort and time (e.g. Belokurov et al., 2003, 2004; Bloom et al., 2012; Brink et al., 2013; Nun et al., 2015; Benavente et al., 2017; Castro et al., 2018). Learning features directly from the data is one way to avoid manual design and can be very useful to find informative attributes for classification (Cabrera-Vives et al., 2016, 2017; Sedaghat & Mahabal, 2017; George & Huerta, 2017; Shallue & Vanderburg, 2018; Charnock & Moss, 2017; Mahabal et al., 2017; Protopapas, 2017). However, even if representative features are obtained, if the data from the pre-processing step is not informative enough or contains errors from the procedure it will be difficult to obtain a good classification.

Deep learning techniques are examples of data-driven solutions extracting features automatically that have proven to be successful in classification problems. Convolutional neural networks (Fukushima, 1980) use spatially correlated data such as images (Krizhevsky et al., 2012; Szegedy et al., 2015) and temporal correlations such as audio (Lee et al., 2009; Abdel-Hamid et al., 2014). Recurrent neural networks, such as those containing Long Short Term Memory units (LSTM, Hochreiter & Schmidhuber, 1997; Gers et al., 1999), have been applied to many natural language processing problems

(Goldberg & Hirst, 2017) like translation (Sutskever et al., 2014) and speech recognition (Graves et al., 2013).

Recently, deep learning has been successfully applied to astronomical problems using convolutional neural networks, for example, for real/bogus separation (Cabrera-Vives et al., 2016, 2017), photometry computation (Kimura et al., 2017), calculation of an image comparable to the difference image (Sedaghat & Mahabal, 2017), gravitational wave detection (George & Huerta, 2017) and exoplanet detection (Shallue & Vanderburg, 2018). Recurrent neural networks have been used for light curve classification (Charnock & Moss, 2017; Mahabal et al., 2017; Protopapas, 2017; Naul et al., 2018).

Recurrent convolutional neural networks are a special type of neural network where convolutional layers are combined with recurrent layers. Usually, a first stage of convolutional layers extract features from the raw data and generate high-level representations in deeper layers, then a second stage of recurrent layers uses the features yielded by the convolutional layers to learn time dependencies. Examples of applications are action recognition in videos (Sainath et al., 2015; Donahue et al., 2017; Zhao et al., 2017a) and speech recognition (Zhao et al., 2017b).

The first contribution of this work is to propose a model to classify variable astronomical objects based on a recurrent convolutional neural network (RCNN), which uses sequences of images directly as inputs. In this way, we ensure that all the information available on the images is fed to the classifier without inducing errors by computing the light curve (e.g. in spurious sources) or the difference images (e.g. in badly convolved images). Our model consists of initial convolutional layers aimed at learning spatial correlations automatically from the images at each epoch, followed by a recurrent layer aimed at learning time dependencies between frames within the sequence of images. To the best of our knowledge, this is the first time that sequences of images are used directly to classify astronomical objects.

The second contribution of this work is to deal partially with the transfer learning problem from synthetic to real-world images. We generate synthetic image sequences that take into account the instrumental and observing conditions of different surveys. Using a simulated dataset is faster and more adaptable to different surveys and classification tasks, compared to collecting real labeled data which is time-consuming and fixed to the specific survey where the images were taken. Furthermore, by randomizing simulations correctly we could generate a virtually infinite number of labeled samples, avoiding the problem of manually labeling a large number of real objects. Simulating a dataset allows us, for example, to create data samples for telescopes that are still under construction such as the LSST. We can also tune the simulation parameters according to a specific classification task and scientific objective.

The structure of the remainder of this article is the following: in Section 2 we describe the process of simulating synthetic images and in Section 3 we present our deep learning framework with our proposed RCNN classifier and a light curve classifier for comparison purposes. In Section 4 we present the classification results obtained with our proposed RCNN model, and compare it with a light curve classifier. In Section 5 we discuss the main implications of this work and in Section 6 we summarize the main conclusions as well as future steps.

2 DATA SIMULATION

In order to train the proposed model, we build a simulated dataset of labeled sequences of images, using the following procedure. First, all the information required to mimic realistic observing conditions which correspond to instrument specifications, observation dates, exposure times and atmospheric conditions is gathered. These parameters are specified in Section 2.1. In this work we simulate observing conditions for the HiTS Survey 2015 on band g (Förster et al., 2016). Next, we simulate light curves based on physical and empirical models, and sample them using the observation dates. The instrument specification, exposure time and atmospheric conditions are used to generate an image for each point in the light curve, and this is done for each simulated light curve. In this way, we produce an irregularly sampled movie of 21x21 pixels for each astronomical object. The simulated image sequence dataset is used to train the proposed RCNN model, which is explained in Section 3. In the next sections we explain the image simulation process in detail.

2.1 Synthetic data simulation parameters

Deep neural network models (and machine learning models in general) are applicable to data following the same distribution of the data used to train them. The simulation process was made adjustable to different observing conditions, so our model can be trained and applied to different instruments. To apply the proposed RCNN classifier to a different survey, we should only gather representative parameters, simulate images and train a new model.

The observing condition parameters are composed of camera and exposure parameters. A camera has a unique list of parameters describing the conversion from photons to digital units. An exposure has a unique list of parameters which describe the time and duration of the exposure, as well as the relevant atmospheric conditions during exposure. These parameters are summarized in Table 1. Herein we simulate data for the HiTS survey, which consists of 50 fields observed with 62 CCD cameras per field, and between 25 and 30 observations per field. In this work, we use empirical observing conditions which are sampled from real observations from HiTS. The typical observing conditions are described in Förster et al. (2016).

The simulated images are produced assuming a given point spread function (PSF), which is sampled from a collection of empirical PSFs, an efficiency of conversion from physical units to analog digital units given by the camera and exposure parameters and the sky level given in the exposure parameters. Next, Poissonian and readout noise are added and the saturation level is applied to the resulting image. More details are given in Section 2.3.

Table 1 Image Simulation Parameters: Camera parameters are constant for a given instrument, but exposure parameters vary in time.

Camera parameters	
Gain [e-/ADU]	Read Noise [e-]
Saturation [ADU]	Pixel Scale [arcsec/pixel]
Exposure parameters	
Date [MJD]	Seeing [pixels]
Airmass	Sky brightness [ADU]
Zero Point [mag]	Filter [g, r, i or z]
Exposure Time [sec]	Limiting magnitude [mag]

2.2 Light curve simulation

We use seven classes of astronomical objects (see Table 2): two non variable (non–variable stars and galaxies) and five variable or transient (RR Lyrae, Cepheids, eclipsing binaries, supernovae and asteroids). Variable sources are simulated in two steps: 1) sampling from either a physical model or empirical data, and 2) adjusting their brightness by sampling from the magnitude distributions described below. In order to sample each type of light curve for a given observation date we used different interpolation methods.

We start by sampling a light curve either from a physical model or from empirical data. Table 2 shows the source of the light curves we sampled from. Supernova redshifts and light curves are obtained from simulations which take into account cosmology and supernova rates, the telescope parameters, and physical models for SNe II from Moriya et al. (2017) and spectrophotometric templates for SN Ia from Hsiao et al. (2007). RRLyrae, Cepheids, and eclipsing binaries were sampled from real data from Sesar et al. (2010), Hartman et al. (2006), and using the LSST Catalog Simulation database (`https://www.lsst.org/scientists/simulations/catsim`, CatSim) respectively. Light curves for non–variable objects were simply simulated as a constant light curve, and asteroids as a single peak. Galaxy simulations are explained in Section 2.3.

We sample light curves using the empirical exposure parameters from HiTS and scale them to follow

[1]`https://www.lsst.org/scientists/simulations/catsim`

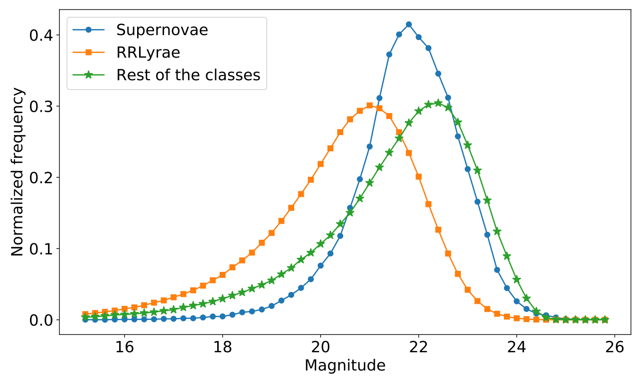


Figure 1. Magnitude density distribution of the simulated data.

Table 2 Class description, astronomical sources simulated in this work.

Astronomical Object	**Generation model**
Supernovae	Simulations based on physical models of SNe II from (Moriya et al., 2017) and SN Ia spectrophotometric templates from (Hsiao et al., 2007)
RR Lyrae	483 light curve templates, sampling a random phase and average magnitude (Sesar et al., 2010)
Cepheids	600 real cepheids light curve Hartman et al. (2006) fitted with a Gaussian process for interpolation (Rasmussen & Williams, 2005)
Eclipsing Binaries	375 Eclipsing binaries templates from CatSim[1], part of the LSST simulation tools
Non-Variable objets	Constant brightness value for each time of observation
Galaxies	Exponential and De Vaucouleur's luminosity profile using parameters from SDSS galaxy catalog (Blanton et al., 2017)
Asteroids	Simulated as a bright source in just a single time of observation

a magnitude distribution that reproduces the HiTS observations. Magnitudes for non–variable sources, eclipsing binaries, Cepheids, and asteroids are sampled from the green curve in Figure 1. This distribution was obtained by fitting an exponential function to the distribution of stars in HiTS. A constraint was added to smooth the decay at large magnitudes in order to follow the supernovae magnitude distribution. This was performed by multiplying the exponential density distribution by a cutoff function $f(m, m_{\mathrm{cutoff}}) = 1 - \mathrm{erf}(m - m_{\mathrm{cutoff}})/2$, where erf is the error function and m_{cutoff} is the value where $f(m, m_{\mathrm{cutoff}}) = 0.5$. The supernovae magnitude distribution decay at large magnitudes can be mimicked by using $m_{\mathrm{cutoff}} = 22.8$ for the all the classes, except for RR Lyrae and galaxies. For RR Lyrae, $m_{\mathrm{cutoff}} = 21.5$ was chosen to make the distribution with magnitude boundaries based on Medina et al. (2018).

2.3 Image simulation using light curves

Having simulated light curves sampled with the corresponding cadence, we use the zero point values $Z_p(t)$ to convert each point of the light curve from magnitudes to ADU using:

$$m(t) = Z_p(t) - 2.5 \log\left(\frac{\mathrm{ADUs}(t)}{T(t)}\right), \qquad (1)$$

where t is the observation time and $T(t)$ is the exposure time for an image at time t. Usually, there are other terms in this conversion associated with airmass and color, but these Z_p's were computed using PanSTARRS1 (Chambers et al., 2016) to fit the resulting magnitude of known sources. For each of the light curves we choose a random CCD array and use its exposure parameters at different epochs. Then, for each point in ADU units of the light curve, a point spread function (PSF) $p_t(x, y)$ is used to generate a source image, where x, y are pixel coordinates and $\sum_{x,y} p_t(x, y) = 1$. We generate the source image (see example shown in Figure 3) by creating an empty image of 21×21 and adding $\mathrm{ADUs}(t) \cdot p_t(x - x_0, y - y_0)$ where x_0, y_0 is the center of the source in the 21×21 image, sampled from a

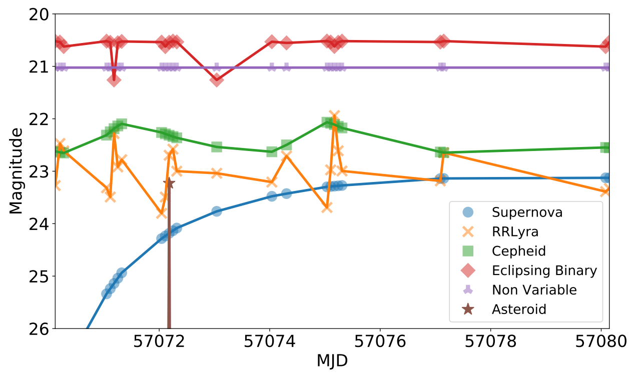


Figure 2. Light curve examples for each class of astronomical object in the database. Galaxies are simulated by using method in section 2.3

uniform distribution within the single image central pixel to simulate random centering errors. The PSF p_t is estimated by averaging real source images from the HiTS survey and computing its FWHM by fitting a 2D-Gaussian function as an estimation of size. We used PSFs estimations with different sizes, but for each observation time t we match one p_t with the current FWHM(t). Dates with FWHMs larger than 2" were not used. A random rotation and mirroring is applied to make the classifier invariant to rotations of the PSF.

Some of the sources may have a host galaxy, so we simulated them by using exponential and De Vaucouleurs profiles with parameters obtained from the Sloan Digital Sky Survey (SDSS, Blanton et al., 2017), including the following (if many bands are used, these quantities are per band): radii, ellipticities, proportion between the two profiles, and the luminosity of the galaxy in magnitudes. The magnitudes are converted to ADUs units and distributed using the exponential + De Vaucouleurs profile. These profiles have a spike in the center, concentrating most of the flux in the central pixel. In order to avoid this issue, we sample 20 uniform random positions inside the central pixel, compute the bulge profile and average them to distribute the flux correctly across the image. We finally convolve this image with $p_t(x, y)$ generating a galaxy image $\mathrm{IM}_{\mathrm{gal}}$. In order to simulate SN in a host galaxy, we sample the position from a distribution following the exponential profile. Figure 3 shows an example image of a simulated SN on a host galaxy.

The last step for image simulation is producing a joint image by adding up the PSF-like image, the galaxy, and the sky brightness $\mathrm{Sky}(t)$ for time t. Then, we convert ADU pixels to electrons $e-$ multiplying by the corresponding Gain of the camera, in order to apply independent Poisson noise to each pixel and Gaussian readout noise. Finally, the image is converted back to ADUs. An example of the resulting image is shown in Figure 3.

Host galaxies were added on 50% of the supernovae objects and 5% on the rest of the classes. These proportions are artificial, but based on prior knowledge about the abundance of supernovae occurrence on hosts galaxies and the low probability of finding other classes near a host galaxy. As mentioned before, the galaxy class is a simulated image of a host galaxy with varying exposure parameters.

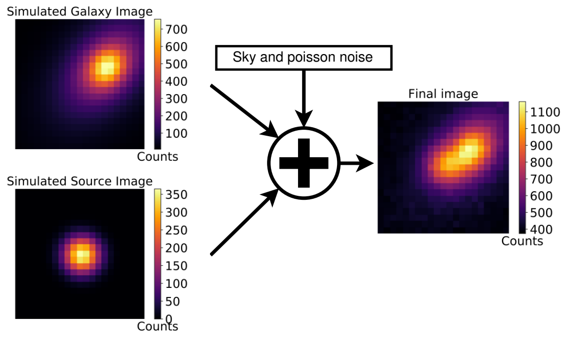


Figure 3. Summary of the image simulation process. The light coming from the source is spread in the Source Image. A simulated host galaxy is added to the source image. The sky brightness is added as a constant value to all the pixels and Poisson noise is sampled with variance equal to the number of photons in each pixel.

Examples comparing real image sequences from HiTS and simulated sequences for non–variable objects can be found in Appendix A.

3 DEEP LEARNING FRAMEWORK

Our goal is to discriminate among the seven classes shown in Table 2. Fig. 2 shows examples of light curves for each of the categories (except for the galaxy class). We simulated 686,000 objects for the training set, 85,750 for validation set and 85,750 for the test set. Each set has a balanced number of objects per class.

We propose a new model to classify astronomical objects based on a recurrent convolutional neural network (RCNN), which uses a sequence of images as input. Convolutional layers are able to automatically learn the spatial correlation between pixels in the input image and extract high-level features, which are used by the recurrent layer to learn time dependencies among images sampled at irregular times. The components of this network are the following:

Fully connected layer Usually placed at the end of the architecture. It takes high-level representations obtained by lower layers and applies the transformation $Y = f(WX + b)$, where X is the input of the layer, W are the weights, b is the bias and f is non-linear activation function. Typical choices for f are the rectifying linear unit (Nair & Hinton, 2010), hyperbolic tangent or logistic sigmoid.

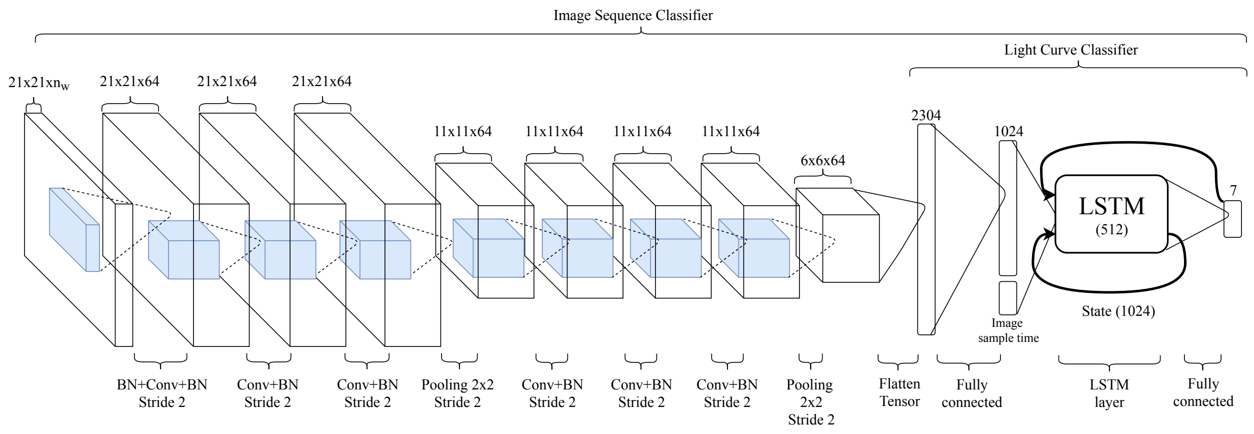


Figure 4. RCNN architecture. The input tensor with shape $(21, 21, n_w)$ is shown at the left of the image. Every layer is described at the bottom and the shape of the data is described at the top. The light curve classifier architecture is the one specified on the right side of the image.

Convolutional layers These layers apply convolutional operations to their inputs. The most popular one is the convolution over images done by filters, which are adjusted through the training process (Fukushima, 1980; Krizhevsky et al., 2012; Szegedy et al., 2015). The convolution Y between an image X with a filter W plus a bias b is expressed as:

$$Y_{i,j,k} = \sum_{m,n,p} X_{i-m,j-n,p} \cdot W_{m,n,p,k} + b_k, \qquad (2)$$

where X is a 3D tensor, the first two coordinates m, n run over the position of the image and the third one p runs over the depth or the number of channels. W has one more dimension k representing the output channels. Right after convolution, a non-linear activation function is used, same as in the fully connected layer.

Table 3 Recurrent Convolutional Neural Network architecture.

Layer	Layer Parameters	Output Dim
Input Layer	$21 \times 21 \times n_w^{a}$	$21 \times 21 \times n_w$
BN (Batch norm)	n_w (mean and std)	$21 \times 21 \times n_w$
Conv + BN	$3 \times 3 \times 64$, 64	$21 \times 21 \times 64$
Conv + BN	$3 \times 3 \times 64$, 64	$21 \times 21 \times 64$
Conv + BN	$3 \times 3 \times 64$, 64	$21 \times 21 \times 64$
Max pooling	2×2, stride 2	$11 \times 11 \times 64$
Conv + BN	$3 \times 3 \times 64$, 64	$11 \times 11 \times 64$
Conv + BN	$3 \times 3 \times 64$, 64	$11 \times 11 \times 64$
Conv + BN	$3 \times 3 \times 64$, 64	$11 \times 11 \times 64$
Max pooling	2×2, stride 2	$6 \times 6 \times 64$
Fully connected (with dropout)	2304×1024	1024
LSTM	1024 + Δt of samples 512 units	512
Output softmax	512×7	7 (n° classes)

[a] n_w is the number of images stacked in the input tensor

Pooling layers These layers are applied to reduce the representation dimensionality through the network and act as a regularizer. In this work, we used max pooling operations that takes the maximum value within a sub-matrix of the image, $Y_{i,j} = \max\left(X_{i:(i+n),j:(j+m)}\right)$ where i, j are the image coordinates and n, m are the coordinates of the sub-matrix.

Batch Normalization It is used to standardize the values of the variables by shifting and scaling them. Some of the effects of batch normalization are better training speed and regularization (Ioffe & Szegedy, 2015). It is normally used between convolutional layers and the operation is:

$$Y = \left(\frac{X - \mu_d}{\sigma_d}\right)\gamma + \beta, \qquad (3)$$

where μ_d and σ_d are the mean and standard deviation of the data computed using a moving average $\mu_d = \alpha\mu_d + (1-\alpha)\mu_{\text{batch}}$, where μ_{batch} is the mean for the current batch, α is a momentum parameter, and γ and β are trainable parameters. In the case of convolutional layer outputs, this operation is applied independently for each channel.

Recurrent Layers and LSTM It is the part of the network that learns time dependencies in a sequence of inputs. There are many types of recurrent layers and networks (Lipton et al., 2015) but most of them have a feedback connection to the input from previous time steps or a state (or both), where a state is an arbitrary representation of a memory of previous inputs. In particular, the recurrent model used in this work is the Long Short Term Memory (LSTM, Hochreiter & Schmidhuber, 1997; Gers et al., 1999). The main characteristic of LSTM is the use of gates that control the content of the state in order to learn longer time dependencies than regular recurrent networks. LSTM has three main gates: the forget gate removes part of the state using the information from current input and previous output, the input gate updates the state, and the output gate combines input, state, and previous output.

3.1 Sequence input to the model

Before inputting an image to the classifier we perform a preprocessing step that consists in subtracting the sky in counts to the image as a constant value. Then, we multiply each pixel by a factor that ensures the same number of counts for a given magnitude, making the conversion in equation 1 invariant to the zero point zp. The operation is the following:

$$\mathrm{IM}_{\mathrm{proc}}(x, y) = (\mathrm{IM}_{\mathrm{orig}}(x, y) - \mathrm{sky}) \cdot 10^{\frac{zp_{\mathrm{ref}} - zp}{2.5}}, \quad (4)$$

where $\mathrm{IM}_{\mathrm{proc}}(x, y)$ is the resulting preprocessed image, $\mathrm{IM}_{\mathrm{orig}}(x, y)$ is the original image, and zp_{ref} is a reference zero point, chosen as the zero point of the first exposure for each field. We use $\mathrm{IM}_{\mathrm{proc}}(x, y)$ to build the inputs to the model.

In the case of supernovae and asteroids, we considered objects where the source is detectable in at least one of the images within the sequence i.e., one point of the original light curve in magnitude must be above the limit of magnitude. We consider the first point when this happens as the “first alert” triggered by the rise of flux in time.

Because we are interested in classifying supernovae in the early stages of the explosion, once an alert is triggered (supernova explosion or asteroid appearance) at time t_i, we query the five images before the alert and create a stack of n_w consecutive images using t_{i-5} as the first image of the stack. In this work we used $n_w = 3$, so if the alert occurs at time t_i, then the input images for the first stack are at time $(t_{i-5}, t_{i-4}, t_{i-3})$, then next time step input will be at $(t_{i-4}, t_{i-3}, t_{i-2})$, then $(t_{i-3}, t_{i-2}, t_{i-1})$ and so on. Therefore the input to the model is a stack of n_w consecutive images creating an input tensor of shape $(21 \times 21 \times n_w)$.

We define N_d as the number of available dates, which corresponds to the number of points between five images before the first detection and last exposure on the respective field. N_d depends on the first detection date. We use the same N_{d} dates to build the input sequence of other classes, in order to have variable size and observation conditions for every class in the dataset. We truncated the maximum number of available dates to $N_d = 20$ for every object to evaluate the model.

We use $n_w > 1$ images stacked at the input and not a single image because convolutional layers can learn part of the short timescale dependencies between these n_w consecutive images, letting the recurrent layer learn about longer timescale dependencies. We also compute the difference in sampling time between the first image of the sequence and the rest of the images, so the model receives information about the irregular time sampling for each image.

3.2 Image sequence classifier architecture

Our model uses high-level representations of the image obtained by convolutional layers as inputs to the recurrent layer. This way, we can add information to the memory of the classifier while the images are received by changing the input as explained above. The LSTM units in the recurrent layer contain memory cells that store learned knowledge from previously seen input images.

As mentioned above, the input is a three dimensional tensor of size $(21, 21, n_w)$, made by n_w consecutive 21×21 images. We first apply a batch normalization layer, followed by a convolutional layer to increase the number of channels from n_w to 64 and an extra batch normalization layer. We use 64 filters and a stride of 1 on each convolutional layer with ReLU as activation function. Also a batch normalization layer at the output of each convolutional layer is implemented shifting and scaling for each channel, a pool layer after the first three convolutional layer + batch normalization, followed by three convolutional layers + batch normalization and a final pool layer.

The output of the pool layer is flattened to a vector of size $6 \times 6 \times 64 = 2304$, i.e. the input to the first fully connected layer with 1024 hidden units. The time difference between the day of observation of the n_w images and the first image of the entire sequence (n_w size vector) is added to the input of the LSTM layer with 512 units. The initial state of the LSTM is an array filled with zeros and the state is updated for every input tensor with n_w stacked images. Finally, the LSTM output is passed through a fully connected layer with softmax activation functions. The details of each layer are shown in Table 3. Fig. 4 shows an illustration of the RCNN architecture.

3.3 Light curve classifier

In order to quantify the advantages of using the image sequence directly for the classification task, we designed a light curve classifier using the fully connected part of the RCNN architecture, shown in Figure 4 and specified by the last 3 rows in Table 3. Since convolutional layers learn the features directly from the images and pass them to the fully connected and recurrent part of the network, we used the latter section of the architecture as a classifier with light curves computed from images as inputs.

The input to the light curve classifier is similar to the one explained in section 3.1. For each of the images at time t_i given to the image sequence classifier, we give the point of the light curve and its variance estimated from the image using optimal photometry (Naylor, 1998). We used $n_w = 3$, which implies a 6 dimensional input (3 points of the light curve and 3 respective errors) for the light curve classifier and also the difference between observation time as described in 3.1. We also applied the flux factor shown in equation 4, which was explained in section 3.1.

3.4 Training Process

Recurrent neural networks are trained using gradient descent (or variations), by using backpropagation through time. Since features are extracted from the image using convolutional layers and are fed to the recurrent layer, the gradients through time are also used to correct the parameters of the convolutional layers. We used cross-entropy as loss function to compare the outputs of the model with the labels. The total loss of a single example has the form $\text{loss} = \sum_t^{N_d} \text{loss}(t)$ where $\text{loss}(t)$ is the cross-entropy at time step t. The training algorithm is Adam (Kingma & Ba, 2014) which is an adaptive learning rate algorithm. The batch size was 256 and we run 30,000 iterations presenting a single batch per iteration to the image sequence classifier and the light curve classifier. The final model was chosen by selecting the one that had the lower loss in the validation set during the training process. We used graphic processor units (GPUs) to train the models. Each model takes 6 hours approximately to complete 30,000 iterations in a GeForce GTX 1080 Ti.

4 RESULTS

After completing the training process, we evaluate the image sequence classifier and the light curve classifier models over the simulated dataset and on real images from HiTS survey with known labels. Five classes are available for real image sequences: supernovae, RR Lyrae, eclipsing binaries, non-variables, and asteroids. We test 80, 111, 76, 500 and 55 examples per class, respectively. In the case of supernovae and asteroids, we define the first detection as the first point where the number of counts on the estimated light curve is five times higher than the error, then build the input to the models as described in section 3.1. For the rest of the classes, the sequence starts at the first exposure. Figure 5 shows a comparison between the image sequence classifier (RCNN) and the light curve classifier, in terms of the evolution of accuracy as the models are fed with more samples, images in case of the RCNN and light curve points in case of light curve classifier, for simulated data and real data. The figure shows also the recall of each of the available classes of real data and accuracy for the simulated dataset as a function of the magnitude of the objects. From Figure 6 to 9, confusion matrices are shown for both the image sequence classifier and light curve classifier, for simulated and real data. The confusion matrices are for time step 20, instant at which most of the samples have already been presented to the classifiers.

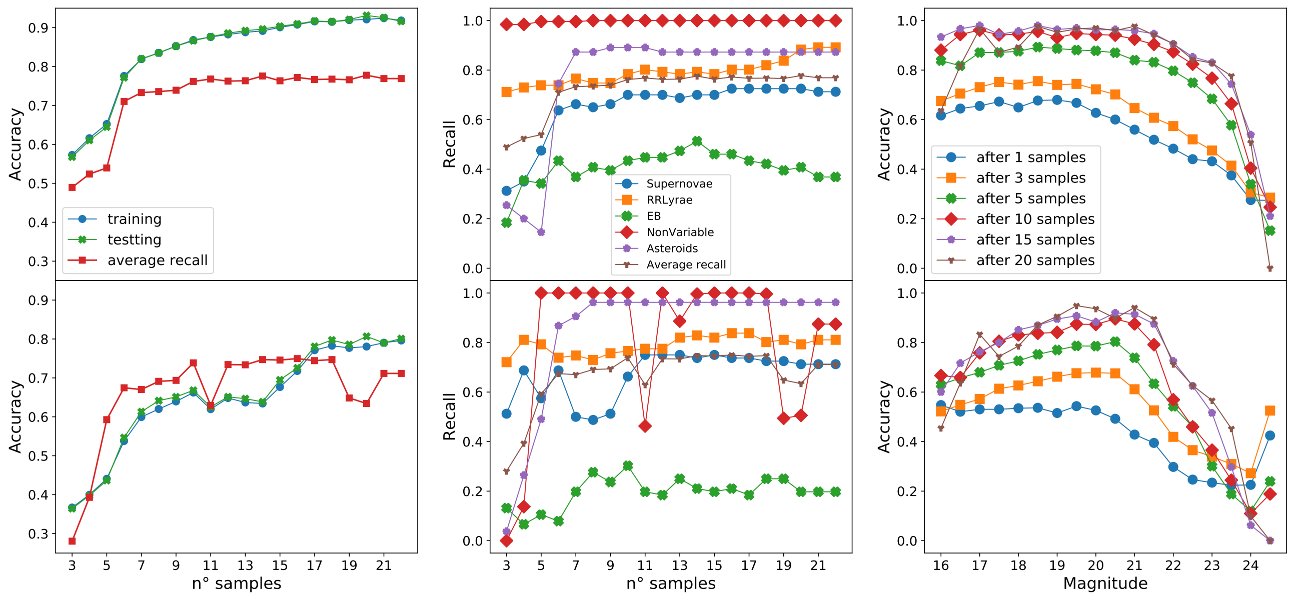
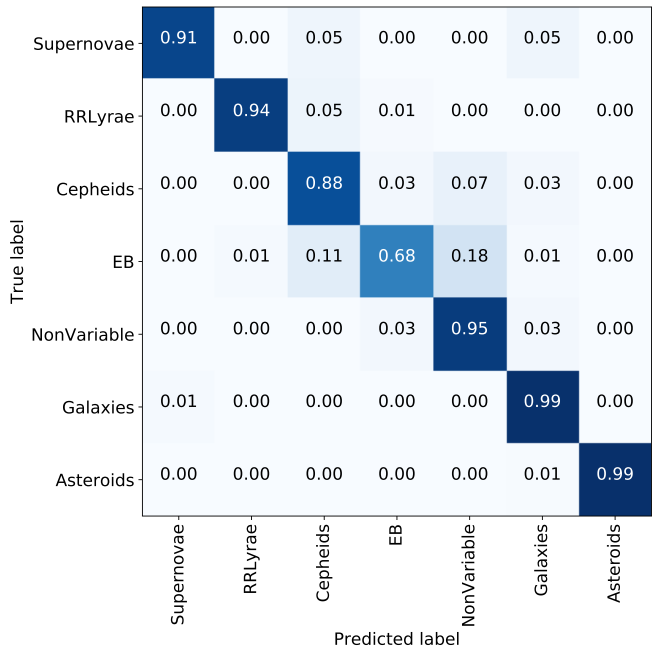


Figure 5. Results comparison between image sequence classifier (top row) and light curve classifier (bottom row). The left plot shows the accuracy for training and testing simulated data, and the average recall for available real classes. The mid plot shows the evolution of recall per class as a function of the number of images (or points) presented to the model. The right plot shows the accuracy on the simulated database as a function of the object magnitude and the number of images.

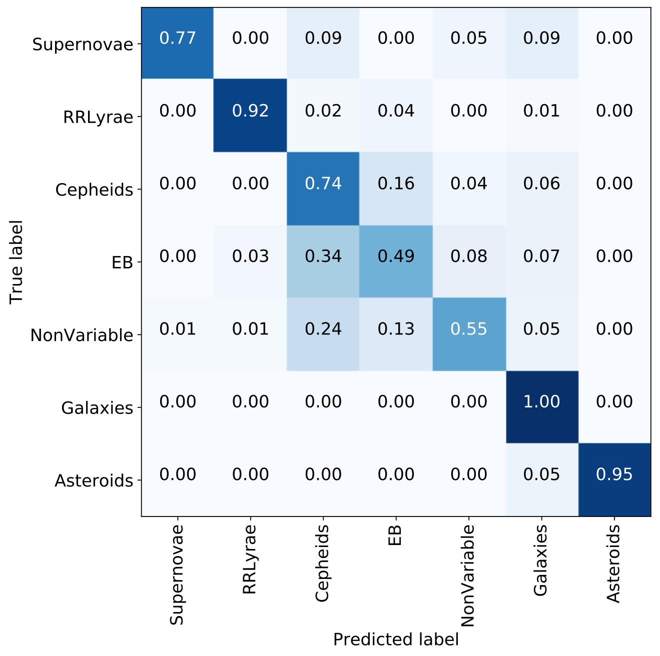
5 DISCUSSION

As can be seen on the left plots of Figure 5, the accuracy on the simulated database and recall on the real database are higher for the image sequence



True label \ Predicted label	Supernovae	RRLyrae	Cepheids	EB	NonVariable	Galaxies	Asteroids
Supernovae	0.91	0.00	0.05	0.00	0.00	0.05	0.00
RRLyrae	0.00	0.94	0.05	0.01	0.00	0.00	0.00
Cepheids	0.00	0.00	0.88	0.03	0.07	0.03	0.00
EB	0.00	0.01	0.11	0.68	0.18	0.01	0.00
NonVariable	0.00	0.00	0.00	0.03	0.95	0.03	0.00
Galaxies	0.01	0.00	0.00	0.00	0.00	0.99	0.00
Asteroids	0.00	0.00	0.00	0.00	0.00	0.01	0.99

Figure 6. RCNN based image classifier confusion matrix on simulated data after 20 inputs (22 images)



True label \ Predicted label	Supernovae	RRLyrae	Cepheids	EB	NonVariable	Galaxies	Asteroids
Supernovae	0.77	0.00	0.09	0.00	0.05	0.09	0.00
RRLyrae	0.00	0.92	0.02	0.04	0.00	0.01	0.00
Cepheids	0.00	0.00	0.74	0.16	0.04	0.06	0.00
EB	0.00	0.03	0.34	0.49	0.08	0.07	0.00
NonVariable	0.01	0.01	0.24	0.13	0.55	0.05	0.00
Galaxies	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Asteroids	0.00	0.00	0.00	0.00	0.00	0.05	0.95

Figure 7. Light curve classifier confusion matrix on simulated data after 20 inputs (22 points of the light curve)

classifier than the light curve classifier. Comparing the confusion matrix on simulated data shown in Figure 6 with the confusion matrix in Figure 7, we found that there are more supernovae misclassified when using the light curve classifier than with the image sequence classifier. Supernovae are misclassified mostly as Cepheids, non–variables and galaxies by the light curve classifier. A possible explanation is that the light curve classifier has no information about the presence of galaxies, whereas the image sequence classifier can infer this from the data. This effect seems to be more important for fainter sources, which tend to be more distant and have smaller angular size galaxies, as can be seen in Figure 5, right plot, where there is a clear improvement in accuracy for fainter sources when using the image sequence classifier in comparison to the light curve classifier.

The image sequence classifier also shows an improvement in the classification of the variable star classes with respect to the light curve classifier in the case of the simulated dataset, as can be seen in Figure 6 and 7, respectively. This result suggests that the image sequence classifier is able to retrieve the necessary information from the image to solve the classification task. Since there is not much bias on each class apart from the light curve shape, the model must learn to perform some form of photometry and extract the flux generated by the object on the image. Thus, we can infer that the image sequence classifier is capable of retrieving the underlying light curve used to simulate the source as well as additional information from the image, such as the presence of a host galaxy.

Another important issue is the effect of giving some historical information about a source (supernovae and asteroids) before the detection. In Figure 5, left and mid plots, we can see the improvement in the accuracy and recall on sample number 6, at the moment of the detection of a supernova or asteroid. In Appendix B we illustrate some examples of the image sequence classifier working on real supernovae, where we can observe this effect.

It is important to mention the fact that the simulations are good enough to classify correctly most of the objects in the HiTS dataset. We can see from Figure 8 that real non–variable objects are perfectly classified using the image sequence classifier, as well as the majority of supernovae (71%), RR Lyrae (89%) and asteroids (87%) in the HiTS dataset, resulting in an average recall of 87% for these four classes. This means that part of the transfer learning problem from simulated to real-world images is solved by the proposed methodology to convert the flux of a light curve to image stamps. The problem reduces to obtaining the right distribution that represents the physical process of the light passing through the atmosphere, lenses and being captured by the CCD camera, by using the estimated exposure conditions, camera parameters and point spread function from empirical data.

Because the image simulation process is the same

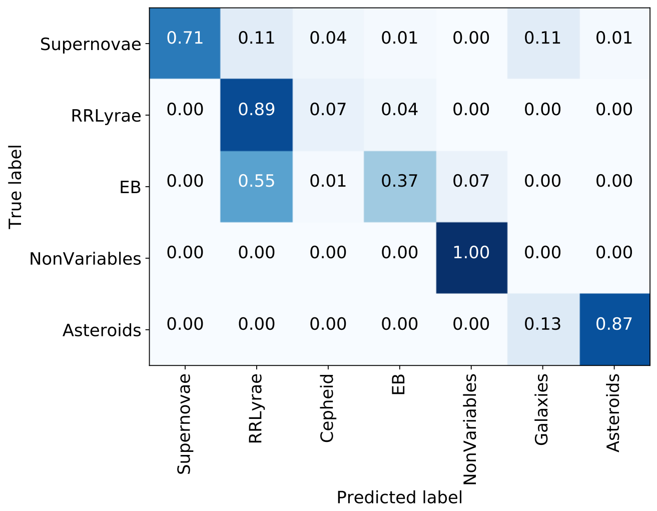


Figure 8. RCNN based image classifier confusion matrix on HiTS real data after 20 inputs (22 images)

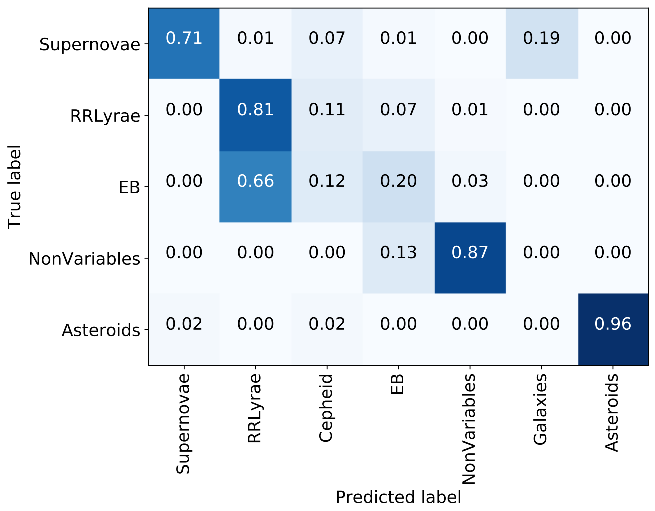


Figure 9. Light curve classifier confusion matrix on HiTS real data after 20 inputs (22 points of the light curve)

for all classes, the fact that eclipsing binaries are misclassified as RR Lyrae resulting in a recall of 37%, suggests that the key feature to fix is the light curve model used to generated synthetic data. The relevant features could be the magnitude distribution, light curve variance as a function of magnitude, or the periods in the case of variable stars. Simulated light curves must be similar enough to the real ones in order to classify them correctly. Furthermore, the relevant features of a simulated object must be similar enough to the real ones, so the distribution for each class such as magnitude, variance, detection criteria, or the proportion of examples with a host galaxy should be chosen carefully according to the scientific goals.

6 CONCLUSIONS

We have proposed a new sequential classification model for astronomical objects based on a convolutional recurrent neural network, which uses directly sequences of images, without computing the light curve or the difference image. Using empirical and astrophysical models of different astronomical objects, we simulated data from the High cadence Transient Survey (HiTS) to generate a synthetic dataset of images for these astronomical objects, considering realistic atmospheric conditions and camera specifications. This synthetic dataset was used to train the image sequence classifier, then we evaluated the model on real images from the HiTS survey. The results show that the proposed model classifies correctly asteroids, RRLyrae, Supernovae and Non–variables from the HiTS survey, with the exception of eclipsing binaries. This has been achieved by using only simulated data for training, the images and observation dates as inputs, and the sky brightness estimation and zero point to pre–process the image. To the best of our knowledge, this is the first time that a sequential classifier using sequences of images as inputs, and without computing the light curve or the difference images, has been proposed in time-domain astronomy.

In order to assess the improvement of using images instead of light curves, we computed optimal photometry on the simulated images and trained a light curve classifier. The light curve classifier has to solve the same classification problem, but using the light curve and the estimated variance. We showed that the proposed sequential classifier outperforms the light curve classifier.

This work also shows that having images at the location of a transient event from before their first detection, as a post-processing step, can be very useful for the classification of astronomical alert streams. For example, in our proposed model having images of a supernova very early rise, before crossing a given threshold on the light curve, can help the classifier report better class probabilities.

A significant part of the transfer learning problem from synthetic to real-world images is solved at the image generation step, which converts flux from a source to an image stamp using realistic considerations. This implies that using synthetic data is a reliable way to train models before acquiring real images from telescopes, as long as we have good light curve models available and the correct parameter distributions to represent well objects for a specific classification task and science goal. The fact that our classifier works well on real-world images after being trained with synthetic data encourage us to use this methodology to train a classifier for new telescopes such as the Large Synoptic Survey Telescope (LSST, Ivezic et al., 2008) and the Zwicky Transient Facility (ZTF, Smith et al., 2014). In this way, we may have a sequential classifier model available before receiving new data.

6.1 Future Work

The results presented in this paper could be improved by searching for the best hyperparameters of the recurrent convolutional neural network, as well as testing different architecture and optimization algorithms. Our model only uses the image sequence preprocessed by zero points and sky brightness, and the dates of observation. Other observation conditions could be added to the model such as using the point spread function as an extra input image, or prior knowledge about the position of an object in the sky. For example, RR Lyrae and Cepheids are more likely to be found in the Milky Way plane than supernovae.

Since we want to simulate other surveys and telescopes, we need to gather the relevant information about the observation conditions and camera specifications. Some telescopes such as the LSST are not operative yet and it is necessary to predict the observation conditions in order to simulate data. For this reason, invariance of the model to observation conditions should be explored to increase robustness to prediction errors. We propose to simulate the sequence of images randomizing observation conditions such as seeing, zero points, or airmass sampled from realistic distributions. Furthermore, our model is currently trained in band g, but it could be adapted to classify the image sequence combining information from more than one band, i.e, a multi-band image sequence classifier.

If a real-world labeled dataset were available, then fine-tuning (Yosinski et al., 2014; Oquab et al., 2014) could be used after training the model with simulated data. Fine-tuning has shown to be a solution for many transfer learning problems. We can also re-adjust the model parameters after training with simulated data to make the classifier capable of solving the problem for real-world data. It is also important to mention that the light curve and image simulation tools developed in this work could be used to test other classifier models.

As future work we propose to modify the eclipsing binary models to improve results on real data and add more astronomical object classes, which means gathering more light curve models and adding them to the simulations, and use better priors for the presence of host galaxies for every object. We can also consider other effects associated with the CCD camera, such as hot pixels and bad columns, incorporating the real/bogus separation naturally into the proposed framework.

7 ACKNOWLEDGEMENTS

Pablo Estevez, Francisco Förster, Pablo Huijse and Guillermo Cabrera-Vives acknowledge support from FONDECYT through grants 1171678, 11130228, 1170305 and 3160747, respectively. Ignacio Reyes gratefully acknowledges financial support from CONICYT-PCHA through its national M.Sc. scholarship 2016 number 22162464. Rodrigo Carrasco Davis gratefully acknowledges financial support from CONICYT-PCHA through its national M.Sc. scholarship 2018 number 22180964. The authors acknowledge support from the Chilean Ministry of Economy, Development, and Tourism's Millennium Science Initiative through grant IC12009, awarded to the Millennium Institute of Astrophysics, MAS. Francisco Förster acknowledge support from Center for Mathematical Modeling through Basal Project PFB-03. Powered@NLHPC: this research was partially supported by the supercomputing infrastructure of the NLHPC (ECM–02). We thank the Nvidia GPU Research Center from University of Chile.

REFERENCES

Abdel-Hamid O., r. Mohamed A., Jiang H., Deng L., Penn G., Yu D., 2014, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22, 1533

Belokurov V., Evans N. W., Du Y. L., 2003, Monthly Notices of the Royal Astronomical Society, 341, 1373

Belokurov V., Evans N. W., Du Y. L., 2004, Monthly Notices of the Royal Astronomical Society, 352, 233

Benavente P., Protopapas P., Pichara K., 2017, The Astrophysical Journal, 845, 147

Blanton M. R., et al., 2017, The Astronomical Journal, 154, 28

Bloom J. S., et al., 2012, Publications of the Astronomical Society of the Pacific, 124, 1175

Brink H., Richards J. W., Poznanski D., Bloom J. S., Rice J., Negahban S., Wainwright M., 2013, Monthly Notices of the Royal Astronomical Society, 435, 1047

Cabrera-Vives G., Reyes I., Förster F., Estévez P. A., Maureira J. C., 2016, in 2016 International Joint Conference on Neural Networks (IJCNN). pp 251–258, doi:10.1109/IJCNN.2016.7727206

Cabrera-Vives G., Reyes I., Förster F., Estévez P. A., Maureira J.-C., 2017, The Astrophysical Journal, 836, 97

Castro N., Protopapas P., Pichara K., 2018, The Astronomical Journal, 155, 16

Chambers K. C., et al., 2016, preprint, (arXiv:1612.05560)

Charnock T., Moss A., 2017, The Astrophysical Journal Letters, 837, L28

Donahue J., Hendricks L. A., Rohrbach M., Venugopalan S., Guadarrama S., Saenko K., Darrell T., 2017, IEEE Transactions on Pattern Analysis and Machine Intelligence, 39, 677

Feast M. W., Menzies J. W., Matsugana N., Whitelock P. A., 2014, Nature, 509, 342

Fukushima K., 1980, Biological Cybernetics, 36, 193

Förster F., et al., 2016, The Astrophysical Journal, 832, 155

George D., Huerta E. A., 2017, Physics Letters B, 778, 64

Gers F. A., Schmidhuber J., Cummins F., 1999, Neural Computation, 12, 2451

Goldberg Y., Hirst G., 2017, Neural Network Methods in Natural Language Processing. Morgan & Claypool Publishers

Graves A., r. Mohamed A., Hinton G., 2013, in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. pp 6645–6649, doi:10.1109/ICASSP.2013.6638947

Hartman J. D., Bersier D., Stanek K. Z., Beaulieu J. P., Kaluzny J., Marquette J., Stetson P. B., Czerny A. S., 2006, Monthly Notices of the Royal Astronomical Society, 371, 1405

Hochreiter S., Schmidhuber J., 1997, Neural computation, 9, 1735

Hsiao E. Y., Conley A., Howell D. A., Sullivan M., Pritchet C. J., Carlberg R. G., Nugent P. E., Phillips M. M., 2007, ApJ, 663, 1187

Ioffe S., Szegedy C., 2015, in Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37. ICML'15. JMLR.org, pp 448–456

Ivezic Z., et al., 2008, ArXiv e-prints arXiv:0805.2366v4,

Kimura A., Takahashi I., Tanaka M., Yasuda N., Ueda N., Yoshida N., 2017, in 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). pp 354–359, doi:10.1109/ICDCSW.2017.47

Kingma D., Ba J., 2014, in International Conference on Learning Representations.

Krizhevsky A., Sutskever I., Hinton G. E., 2012, in Pereira F., Burges C. J. C., Bottou L., Weinberger K. Q., eds, , Advances in Neural Information Processing Systems 25. Curran Associates, Inc., pp 1097–1105

Lee H., Pham P., Largman Y., Ng A. Y., 2009, in Bengio Y., Schuurmans D., Lafferty J. D., Williams C. K. I., Culotta A., eds, , Advances in Neural Information Processing Systems 22. Curran Associates, Inc., pp 1096–1104

Lipton Z. C., Berkowitz J., Elkan C., 2015, ArXiv e-prints arXiv:1506.00019

Mahabal A., Sheth K., Gieseke F., Pai A., Djorgovski S. G., Drake A., Graham M., the CSS/CRTS/PTF Collaboration 2017, in 2017 IEEE Symposium Series on Computational Intelligence (SSCI). pp 1–8, doi:10.1109/SSCI.2017.8280984

Medina G. E., et al., 2018, The Astrophysical Journal, 855, 43

Moriya T. J., et al., 2017, Monthly Notices of the Royal Astronomical Society, 466, 2085

Nair V., Hinton G. E., 2010, in Proceedings of the 27th International Conference on International Conference on Machine Learning. ICML'10. Omnipress, USA, pp 807–814

Narayan G., et al., 2018, The Astrophysical Journal Supplement Series, 236, 9

Naul B., Bloom J. S., Pérez F., van der Walt S., 2018, Nature Astronomy, 2, 151

Naylor T., 1998, Monthly Notices of the Royal Astronomical Society, 296, 339

Ngeow C., Gieren W., Klein C., 2013, Proceedings of the International Astronomical Union, 9, 123–128

Nun I., Protopapas P., Sim B., Zhu M., Dave R., Castro N., Pichara K., 2015, ArXiv e-prints arXiv:1506.00010,

Oquab M., Bottou L., Laptev I., Sivic J., 2014, in 2014 IEEE Conference on Computer Vision and Pattern Recognition. pp 1717–1724, doi:10.1109/CVPR.2014.222

Protopapas P., 2017, in American Astronomical Society Meeting Abstracts #230. p. 104.03

Rasmussen C. E., Williams C. K. I., 2005, Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press

Riess A. G., et al., 1998, The Astronomical Journal, 116, 1009

Sainath T. N., Vinyals O., Senior A., Sak H., 2015, in 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp 4580–4584, doi:10.1109/ICASSP.2015.7178838

Schmidt B. P., et al., 1998, The Astrophysical Journal, 507, 46

Sedaghat N., Mahabal A., 2017, ArXiv e-prints arXiv:1710.01422

Sesar B., et al., 2010, The Astrophysical Journal, 708, 717

Shallue C., Vanderburg A., 2018, The Astronomical Journal, 155, 94

Smith R. M., et al., 2014, Proc. SPIE 9147, Ground-based and Airborne Instrumentation for Astronomy V

Sutskever I., Vinyals O., Le Q. V., 2014, in Ghahramani Z., Welling M., Cortes C., Lawrence N. D., Weinberger K. Q., eds, , Advances in Neural Information Processing Systems 27. Curran Associates, Inc., pp 3104–3112

Szegedy C., et al., 2015, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp 1–9, doi:10.1109/CVPR.2015.7298594

Yosinski J., Clune J., Bengio Y., Lipson H., 2014, in Proceedings of the 27th International Conference on Neural Information Processing Systems. MIT Press, pp 3320–3328

Zhao R., Ali H., van der Smagt P., 2017a, in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp 4260–4267, doi:10.1109/IROS.2017.8206288

Zhao Y., Jin X., Hu X., 2017b, in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp 5300–5304, doi:10.1109/ICASSP.2017.7953168

APPENDIX A: SIMULATION EXAMPLES

In Figs.10 to 12, we show examples of simulated images compared to the real ones. Given a real non–variable source with a known magnitude, we use the same dates where the source was observed and the observation conditions to simulate a non–variable source. Sample time goes from left to right, top to bottom. Since the estimated point spread function used for simulations is computed by averaging single PSFs, simulated images have bigger PSFs than real ones. As can be observed, the pixel distribution on the simulated images is very close to the real ones.

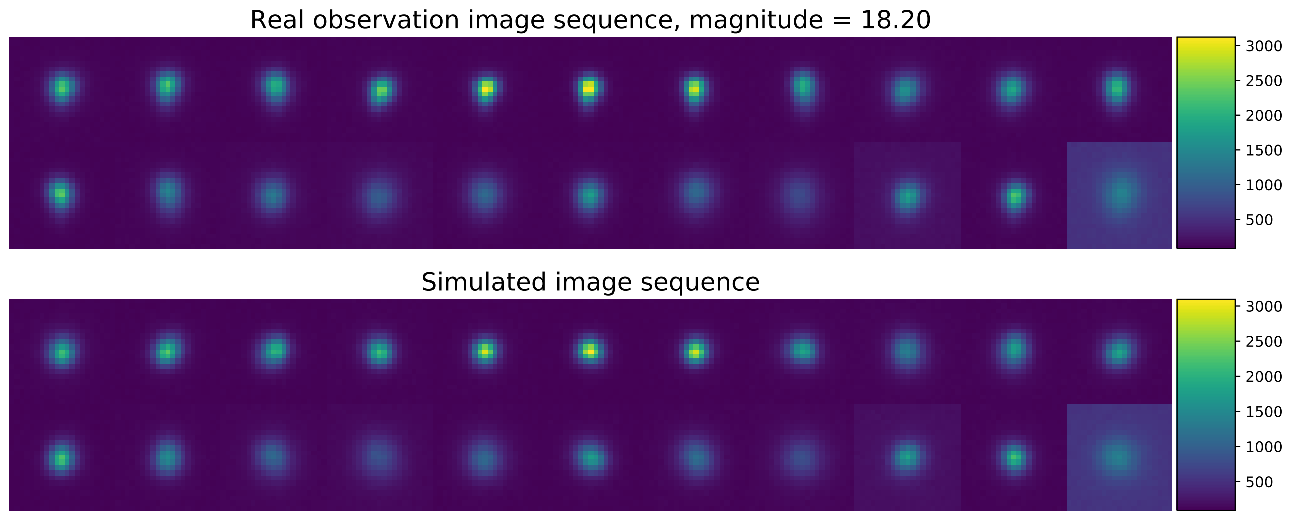


Figure 10. Image simulation example 1

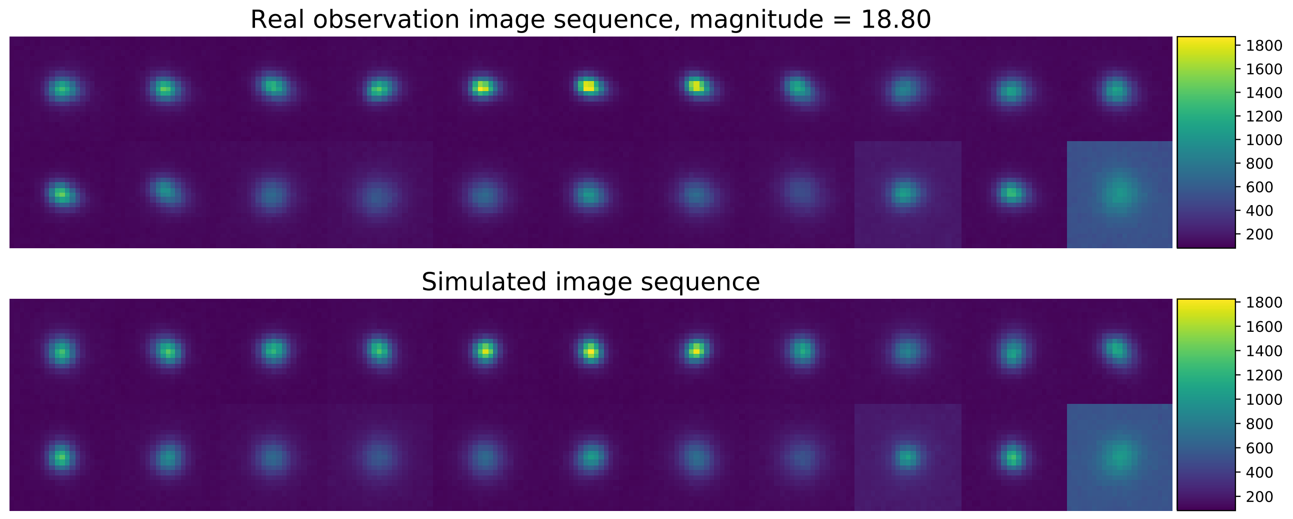


Figure 11. Image simulation example 2

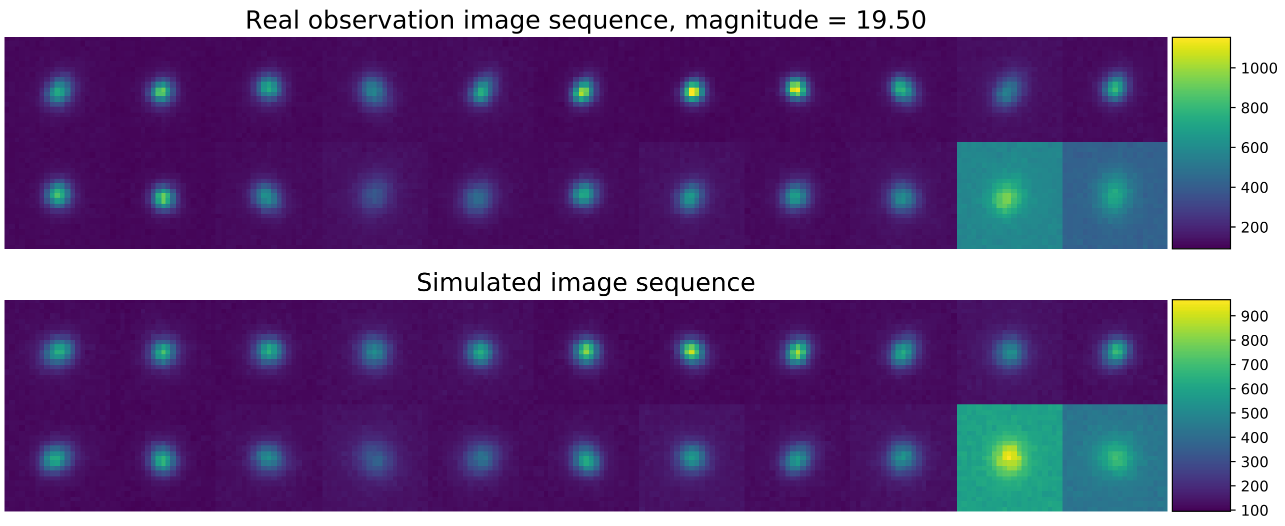


Figure 12. Image simulation example 3

APPENDIX B: EXAMPLES OF THE IMAGE SEQUENCE CLASSIFIER MODEL WORKING ON REAL SUPERNOVAE

Herein we show some examples of the RCNN based image sequence classifier working on HiTS supernovae examples. From Figure 13 to 15, we show the light curve in counts of a supernova and the first detection time (upper plot), probabilities of the object of being of a certain class according to the model through time (mid plot) and the stamps corresponding to each observation date used as input (bottom plot). As mentioned in section 4, the performance of the classifier gets better around the first detection of the supernovae (image number six).

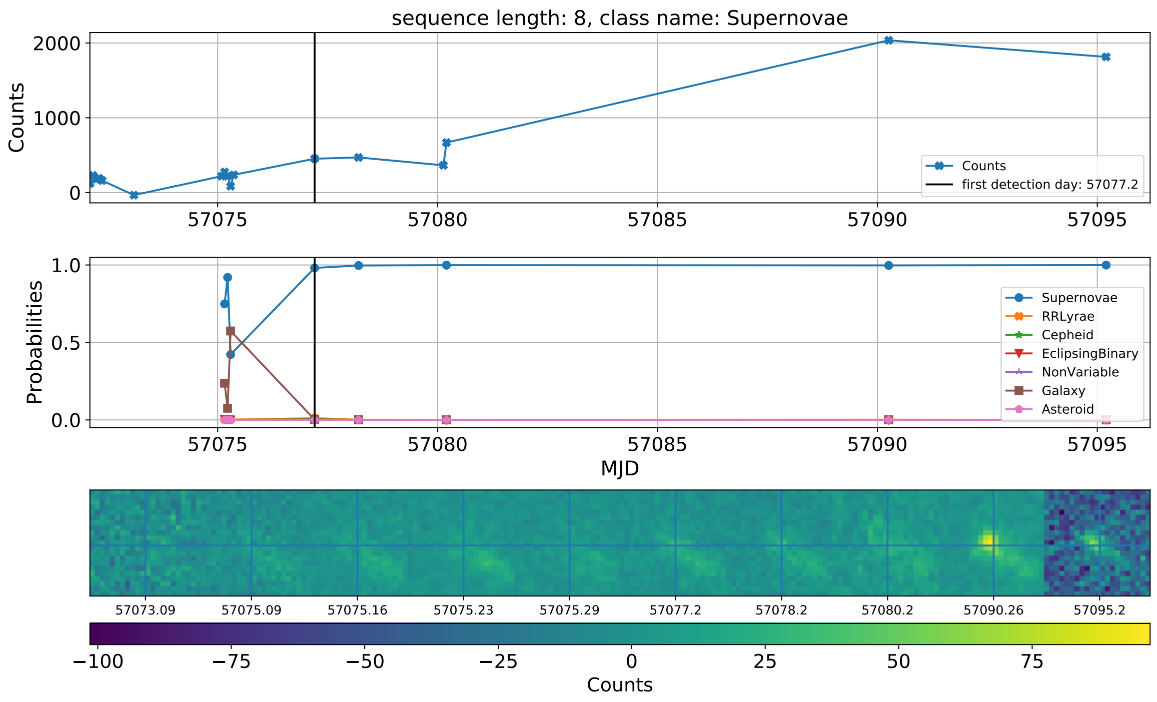


Figure 13. Image sequence classification example 1

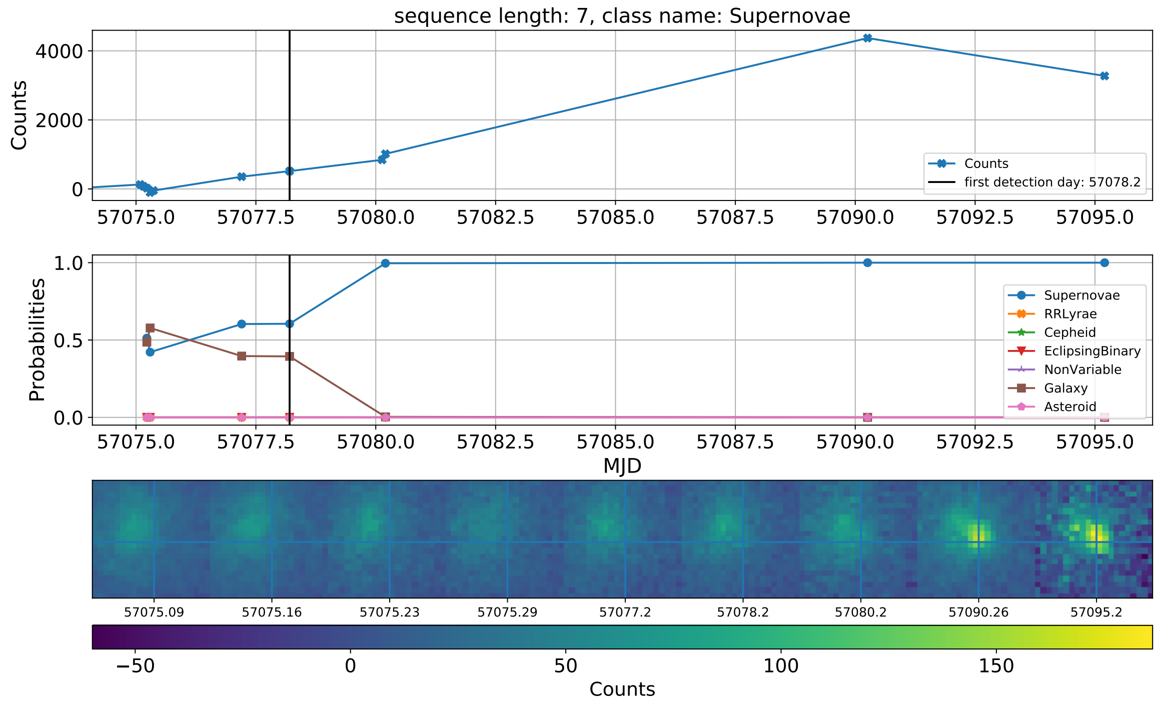


Figure 14. Image sequence classification example 2

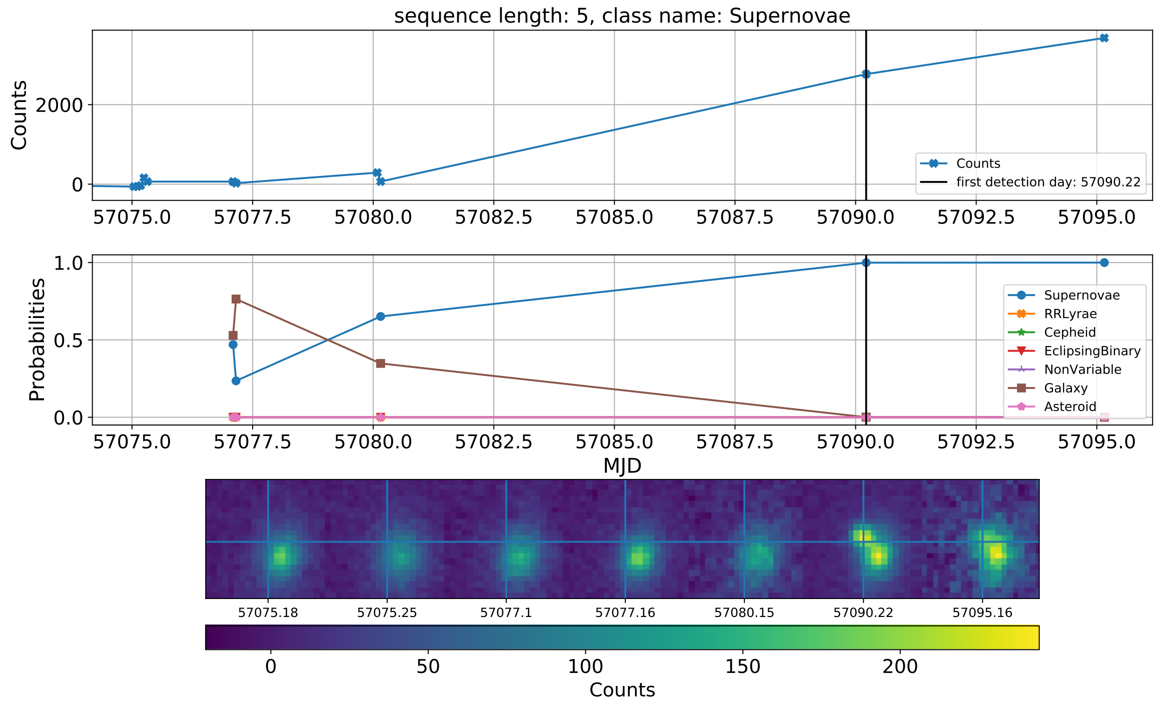


Figure 15. Image sequence classification example 3